**电子科技大学**

**计算机科学与工程学院**

**标 准 实 验 报 告**

**（实验）课程名称 软件开发综合实验**

**电子科技大学教务处制表**

**电 子 科 技 大 学**

# 实 验 报 告

**学生姓名：覃翊 学 号： 2016060107008 指导教师：王丽杰**

**实验地点： 主楼B412 实验时间：11.09-11.16**

**一、实验室名称： 计算机学院实验中心A2-412**

**二、实验项目名称：电话号码本**

**三、实验学时：8学时**

**四、实验原理：**

**通过c语言文件操作结合数组，用哈希算法实现数据的存储，查找以及压缩功能，**

**五、实验目的：**

**设计系统，增加编程能力，考验调试能力，学习常见的哈希算法，调试冲突解决策略**

**六、实验内容：**

**实现一个信息管理系统，具备增加，删除，查找，修改的功能，具有从文件导入，导出至文件的功能，用哈希算法实现快速查找，具备冲突解决策略。能根据姓名快速查找出对应的电话以及地址，若查询失败，则询问是否将此人及其数据加入数据库中。**

1. **实验步骤：**

**首先，我先编写了界面，创建了一个mode变量，用来控制程序的过程以及结束，创建了一个while循环，mode为0则跳出，mode为1则程序进行。这样能循环显示界面，无限地操作。在while循环里，我先显示菜单，然后用户输入一个指令，用switch函数来识别用户输入的指令。**

**当用户输入1时，屏幕显示我预先说明的说明书，包含有对本系统的使用，全文如下：操作1与操作2二选一打开电话本，每次需打开电话本后才能进行操作，本电话本系统支持多电话本管理，不同的文件名对应不同的电话本，打开电话本成功后会得到提示，此时可进行列表所有操作，需要注意的是，当一切操作 结束后，需执行操作4以保存，这样才能从内存将数据保存至文件，本系统高效，稳定，祝使用愉快！**

**当用户输入2时，会新建一个文件，用fopen函数，fp = fopen(listname, "w+"); 参数为w+，是写入文件操作，若没有此文件则创建此文件。打开文件后，指针保留，方便之后的操作。**

**printf("请输入电话本名字：");**

**scanf("%s", listname);**

**fp = fopen(listname, "w+");**

**printf("新建成功，请继续操作。\n");**

**当用户输入3时，打开对应名称的文件，存入已有的数组，然后立刻关闭文件，周所有的操作都在内存中进行了。**

**printf("请输入要打开的电话本名字：");**

**scanf("%s", listname);**

**fp = fopen(listname, "r");**

**store(fp);**

**printf("打开电话本成功，请继续您的操作。\n");**

**fclose(fp);**

**当用户输入4时，会将数组内的所有内容导入文件，这个文件并不是之前打开的文件，而是源文件的补充，重新抹掉之后再把所有的数据导入。**

**fp = fopen(listname, "w+");**

**for(i = 0; i < 1000; ++i){**

**if(Hflag[i] >= 1){**

**fprintf(fp, "%d\t%d\t%s\t%s\t%s\n", Hnumb[i], Hflag[i], name[i], numb[i], addr[i]);**

**}**

**}**

**fclose(fp);**

**printf("保存成功！\n");**

**当用户输入5时，会从数据库中查找，具体步骤是先把名字用哈希算法得到一个基础值，用这个值进行查找，若这个值对应的数组内容就是它本身，就输出，否则用冲突处理函数，用平方再散列法寻找，因为我设置的哈希函数就是用平方再散列法解决的冲突。**

**printf("要显示联系人的姓名：");**

**scanf("%s", nametmp);**

**Hnumbtmp = SearchHTable(nametmp);**

**if(Hnumbtmp > 0){**

**printf("姓名：%s\t电话:%s\t地址：%s\n", name[Hnumbtmp], numb[Hnumbtmp], addr[Hnumbtmp]);**

**}else if(Hnumbtmp == -1){**

**printf("查找失败\n是否添加此人信息？按1回车确认，否则按0\n");**

**scanf("%d", &tmp);**

**if(tmp){**

**Hnumbtmp = GetHkey(nametmp);**

**Hnumbtmp = CollisionKey(Hnumbtmp);**

**Hflag[Hnumbtmp] = 1;**

**strcpy(&name[Hnumbtmp][0], nametmp);**

**printf("请输入该联系人住址：");**

**scanf("%s", addr[Hnumbtmp]);**

**printf("请输入联系人电话：");**

**scanf("%s", numb[Hnumbtmp]);**

**}**

**}**

**break;**

**当用户输入6时，会新建联系人，我先当用户输入要新建的联系人，然后先判断该名称是否在数据库里，判断的具体方法是先取得该名字对应的哈希值，然后对应的冲突处理，以确认该值的确对应的是该名字，然后把这个值对应的flag是否有标记，如果发现这个字符串确认有标记了，意思就是该名字已经在数据库里了。如果名字已经在数据库里了，则询问是否覆盖，然后让用户输入数据进数组，标记flag=1，如果不覆盖，在名字后添加编号以表示序号。如果名字不在数据库，则照常新建输入数据。**

**printf("请输入姓名：");**

**scanf("%s", nametmp);**

**if(Hflag[CollisionKey(GetHkey(nametmp))] == 0){**

**Hnumbtmp = GetHkey(nametmp);**

**Hnumbtmp = CollisionKey(Hnumbtmp);**

**Hnumb[Hnumbtmp] = Hnumbtmp;**

**Hflag[Hnumbtmp] = 1;**

**strcpy(&name[Hnumbtmp][0], nametmp);**

**printf("请输入该联系人住址：");**

**scanf("%s", addr[Hnumbtmp]);**

**printf("请输入联系人电话：");**

**scanf("%s", numb[Hnumbtmp]);**

**}else{**

**printf("该联系人已存在，是否覆盖？按1回车确认，否则按0\n");**

**scanf("%d", &tmp);**

**if(tmp){**

**Hnumbtmp = GetHkey(nametmp);**

**Hflag[Hnumbtmp] = 1;**

**Hnumb[Hnumbtmp] = Hnumbtmp;**

**strcpy(&name[Hnumbtmp][0], nametmp);**

**printf("请输入该联系人住址：");**

**scanf("%s", addr[Hnumbtmp]);**

**printf("请输入联系人电话：");**

**scanf("%s", numb[Hnumbtmp]);**

**}else{**

**char plus[2] = {'\*'};**

**strcat(nametmp, plus);**

**Hnumbtmp = GetHkey(nametmp);**

**Hnumbtmp = CollisionKey(Hnumbtmp);**

**Hflag[Hnumbtmp] = 1;**

**Hnumb[Hnumbtmp] = Hnumbtmp;**

**strcpy(&name[Hnumbtmp][0], nametmp);**

**printf("请输入该联系人住址：");**

**scanf("%s", addr[Hnumbtmp]);**

**printf("请输入联系人电话：");**

**scanf("%s", numb[Hnumbtmp]);**

**}**

**}**

**printf("添加成功！\n");**

**当用户输入7时，可以修改联系人，就是修改各个数组的数据，直接重新覆盖入原来的数组就好了。**

**printf("要修改联系人的姓名：");**

**scanf("%s", nametmp);**

**Hnumbtmp = SearchHTable(nametmp);**

**printf("新电话号码：");**

**scanf("%s", numb[Hnumbtmp]);**

**printf("新住址：");**

**scanf("%s", addr[Hnumbtmp]);**

**printf("修改成功");**

**当用户输入8时，删除联系人，直接把flag置为0就好了，这样在程序中所有的判断都不会认为这里有数据，导入文件的时候直接就抛掉了，使用到查找的算法里最后一步都是判断flag,所以flag置为0就直接删除 掉数据了。**

**printf("请输入要删除联系人的名字：");**

**scanf("%s", nametmp);**

**DeleteHTable(nametmp);**

**printf("删除成功！\n");**

**break;**

**当用户输入9时，显示联系人信息，直接判断flag值，值大于等于1就显示出来，大于等于1的flag值有1 ，2 ，3. 2对应的是亲情联系人，3对应的是黑名单，黑名单不是删除，所以在显示的时候也会显示出来。**

**printf("请输入要设为亲情联系人的名字：");**

**scanf("%s", nametmp);**

**Hnumbtmp = SearchHTable(nametmp);**

**Hflag[Hnumbtmp] = 2;**

**printf("设置成功！\n");**

**当用户输入10时，设为亲情联系人就把flag置为2.**

**当用户输入11时，显示flag等于2的联系人，即亲情联系人。**

**当用户输入12时，设为黑名单就把flag置为2.**

**当用户输入12时，移除出黑名单就把flag置为1.即把该人设为普通联系人。**

**当用户输入14时，显示flag等于3的联系人，即黑名单联系人。**

**当用户输入15时，mode等于0，因为mode是循环控制的把手，所以mode=0就可以跳出循环，结束程序。**

**取得哈希值的函数，我把字符串对应的ascii码加起来，然后模大于1000的最小素数，获得哈希值。**

**int GetHkey(char \*q){//根据姓名求出哈希地址**

**char \*p = q;**

**int itmp;**

**int sum = 0;**

**while(\*p != '\0'){**

**itmp = \*p;**

**sum += itmp;**

**p++;**

**}**

**return sum % 1009;**

**}**

**冲突处理办法，我用了平方再散列法解决，分别试正负1的平方，正负2的平方，以此类推，在查找时我也是用同样的方法试int CollisionKey(int Hnumbtmp){ //冲突处理**

**int i = 1;**

**while(Hnumb[Hnumbtmp] != 0){**

**if(Hnumb[Hnumbtmp + i\*i] == 0){**

**Hnumbtmp += i\*i;**

**}else if(Hnumb[Hnumbtmp - i\*i] == 0){**

**Hnumbtmp -= i\*i;**

**}**

**i++;**

**}**

**return Hnumbtmp;**

**}**

**查找，我读入姓名，然后通过上面的函数获得哈希值，然后查找，也是使用平方再散列法查找，最多查到正负15的平方，这样就查找了30次，很多了，如果查15次找不到就认为该数据库里无此人。当然，如果数据库很大，重复的哈希值可能性很高的话，可以提高数值，但是如果像本例一样没那么高的话，提高数值，获得所谓的“保证”也只是在浪费性能而已。**

**int SearchHTable(char \*nametmp ){**

**int tmp = GetHkey(nametmp);**

**int i = 1, j = 0;**

**while(strcmp(name[tmp], nametmp)){**

**if(strcmp(name[tmp + i\*i], nametmp) == 0){**

**tmp += i\*i;**

**}**

**if(strcmp(name[tmp - i\*i], nametmp) == 0){**

**tmp -= i\*i;**

**}**

**if(++j > 15){**

**return -1;**

**}**

**++i;**

**}**

**return tmp;**

**}**

**删除数据，我直接把flag置零，之后有数据的话就直接覆盖，如果没有数据的话，在写进文件保存的时候这一串数据会自动抛掉，反正这是最简单删除数据的办法，最简单也最高明，并且，更为高明的是可以恢复数据，如果还没保存的话，法线误删了，可以直接跳过flag，不判断，找到还在内存里的文件。**

**void DeleteHTable(char \*nametmp){**

**Hnumbtmp = SearchHTable(nametmp);**

**Hflag[Hnumbtmp] = 0;**

**}**

**展示数据库，flag大于等于1的都能输出，不管是亲情联系人还是黑名单 ，因为黑名单只是拉黑，并没有删除，所以存在黑名单的意义就是在电话号码拨进来的时候进行判断，如果在黑名单里就直接挂断，如果删除数据就没有这个功能**

**void DisplayHTable(){**

**int i;**

**for(i = 0; i < 1000; ++i){**

**if(Hflag[i] >= 1){**

**printf("\t姓名：%12s\n\t电话：%12s\n\t住址：%12s\n\n", name[i], numb[i], addr[i]);**

**}**

**}**

**}**

**存储文件进内存，就是把哈希值，和flag，以及信息放进内存**

**void store(FILE \*fp){**

**while(!feof(fp)){**

**fscanf(fp, "%d", &Hnumbtmp);**

**fscanf(fp, "%d %s %s %s\n",&Hflag[Hnumbtmp], name[Hnumbtmp], numb[Hnumbtmp], addr[Hnumbtmp]);**

**// fscanf(fp, "%d %d %s %s %s\n", &Hnumbtmp, &Hflag[Hnumbtmp], name[Hnumbtmp], numb[Hnumbtmp], addr[Hnumbtmp]);**

**// printf("%d %d %s %s %s\n",Hnumbtmp, Hflag[Hnumbtmp], name[Hnumbtmp], numb[Hnumbtmp], addr[Hnumbtmp]);**

**Hnumb[Hnumbtmp] = Hnumbtmp;**

**}**

**}**

**完整代码我就不附在文件里了，太长，加上了有28页，我已发给老师，当然，我也放在了我的GitHub上：https://github.com/qinyi0411/Software-experiment-course/blob/master/first/%E7%94%B5%E8%AF%9D%E6%9C%AC.cpp**

**八、总结及心得体会：**

通过本实验的练习，学会了文件操作，指针，数组等，实现了很多功能，锻炼了调试能力，程序无bug，以及做出一个完整无bug的管理系统，非常的有成就感。增加了对c语言的熟悉度，锻炼了编程思想。

**九、对本实验过程及方法、手段的改进建议：**

有一个遗憾就是没有加入查电话的功能的，明明超级容易实现，一个for内一个strcmp就完成了，如果相同，就得到了对应的哈希值，然后就把信息输出就好了，非常简单，但是由于程序已经提交了，所以添加不了，很遗憾。本程序是用c语言写的，可以再用c++语言翻译一遍，我会在接下来的时间里用c++复写一次，并加上新功能。

**报告评分：**

**指导教师签字：**